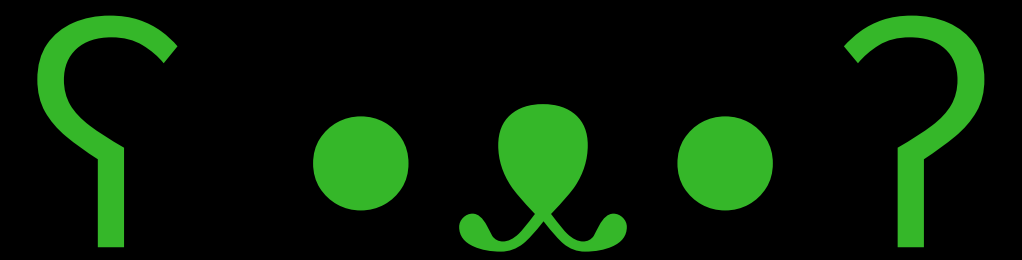


b34r5hell

Web Security

Agenda

> Web CTFs
> Web Basics
> HTTP Protocol
> BurpSuite
> Example CTFs

Prerequisite Knowledge (not much!)

- One of the better categories for beginners
  - Not much math (cryptography) or systems knowledge (pwn, reverse) needed
  - Online documentation (e.g. MDN Web Docs) is your friend!
- Good to have general knowledge of how a web browser works
  - What are each of the tabs in Developer Tools (Inspect Element) for?
- Comfortable working with HTTP requests
  - Editing request headers, data, cookies, etc.
- Knowledge of JavaScript “frameworks” (e.g. React, Vue) is usually not necessary
  - Challenges usually more tailored to exploiting the “backend”

Web Security and CTFs

- Usually given a link to a website hosted on the CTF server
- Must exploit the site in some way to find the flag
- **Possible Attack Methods**
  - Hidden in plain sight (view source/inspect-element)
  - Sending custom requests (e.g. BurpSuite)
  - Manipulating cookies/sessions
  - Cross-Site-Scripting (XSS)
  - SQL Injection
  - XML External Entity
  - Much more

Follow Along

- Similar to last time, we will be going over the exercises under web-security in the Bootcamp Github
- Should only need a browser and the Docker setup

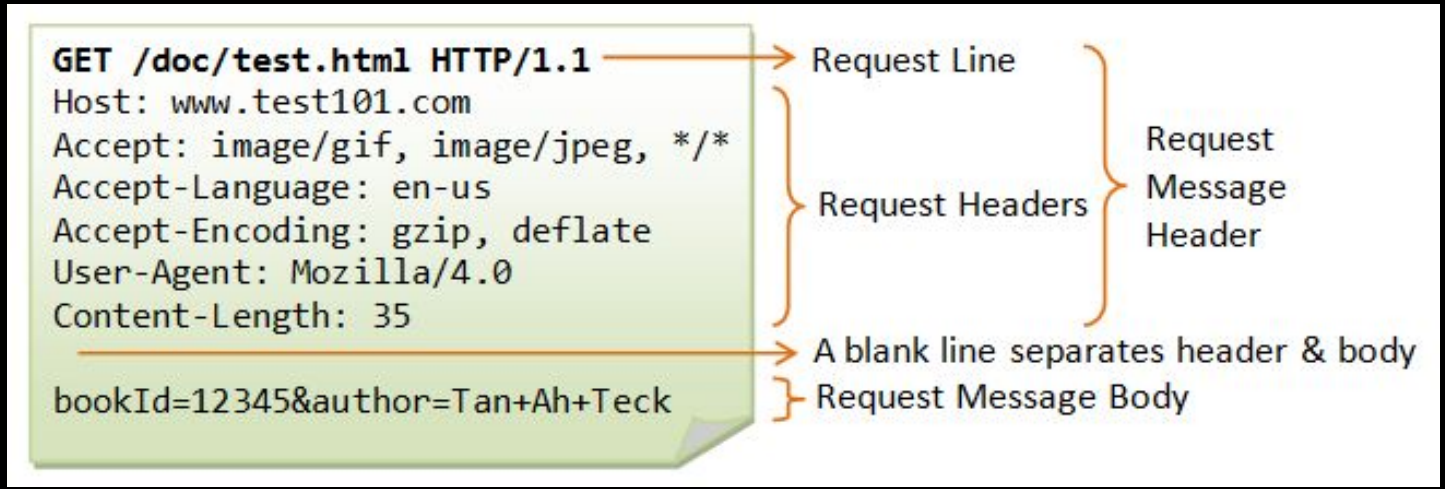
Web Basics

Web Browsers (Client)
> Make requests to servers, receive responses
> Renders **HTML/CSS** and executes **Javascript** code
> Content Security Policy (CSP)
> Stores and handles **cookies**
> Examples: Firefox, Safari, Chromium, etc.

Developer Tools
> Accessed via right-click + “Inspect Element”
> View/manipulate HTML/CSS/JS (client-side only)
> Monitor ingoing/outgoing network traffic
> Manage cookies and other site data

HTTP Protocol

> Request Line = METHOD + SP + REQUEST-URI + SP + HTTP VERSION + CRLF
> Headers allow you to specify additional information
    > Set-Cookie and Cookie headers (and others) allow for **cookies**
> HTTP is a stateless protocol



Sending Data Over HTTP

- POST requests use the HTTP body
- The format of the HTTP body is specified by the **Content-Type header**
- GET requests can specify data via **query parameters**
- Characters like space have a special meaning in HTTP, so **URL encoding** is used: “ “ -> %20
  - % followed by hexadecimal representation of the byte

Curl

> Command-line tool to manage HTTP communication
> Can specify most details of an HTTP request
> Can even act on your behalf and keep track of cookies and more!

ʕ•ᴥ•ʔ

BurpSuite

> Useful tool to capture and manipulate HTTP requests and responses

Tasks

- Download BurpSuite
  - This can be done on either the Linux environment or your computer
- Complete the pwn.college dojo challenges